



Bioinspired computation in combinatorial optimization - Algorithms and their computational complexity

Neumann, Frank; Witt, Carsten

Published in:

Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation

Link to article, DOI:

[10.1145/2464576.2466738](https://doi.org/10.1145/2464576.2466738)

Publication date:

2013

Document Version

Publisher's PDF, also known as Version of record

[Link back to DTU Orbit](#)

Citation (APA):

Neumann, F., & Witt, C. (2013). Bioinspired computation in combinatorial optimization - Algorithms and their computational complexity. In *Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation* (pp. 567-590). Association for Computing Machinery.
<https://doi.org/10.1145/2464576.2466738>

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity

Frank Neumann¹ Carsten Witt²

¹The University of Adelaide
cs.adelaide.edu.au/~frank

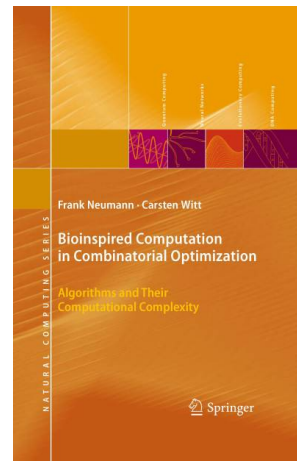
²Technical University of Denmark
www.imm.dtu.dk/~cawi

Tutorial at GECCO 2013

Copyright is held by the author/owner(s).

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.

ACM 978-1-4503-1964-5/13/07.



Book available at www.bioinspiredcomputation.com 1/88

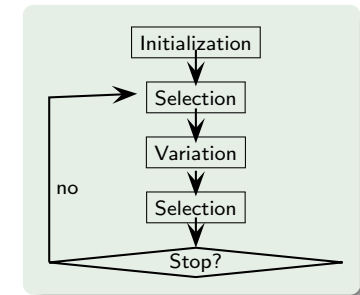
Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Evolutionary Algorithms and Other Search Heuristics

Most famous search heuristic: **Evolutionary Algorithms (EAs)**

- a bio-inspired heuristic
- paradigm: evolution in nature, "survival of the fittest"
- actually it's only an algorithm, a **randomized search heuristic (RSH)**



- Goal: optimization
- Here: discrete search spaces, combinatorial optimization, in particular pseudo-boolean functions

Optimize $f: \{0, 1\}^n \rightarrow \mathbb{R}$

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization 2/88

Why Do We Consider Randomized Search Heuristics?

- Not enough resources (time, money, knowledge) for a tailored algorithm
- Black Box Scenario $x \rightarrow \boxed{} \rightarrow f(x)$ rules out problem-specific algorithms
- We like the simplicity, robustness, ... of Randomized Search Heuristics
- They are surprisingly successful.

Point of view

Want a solid theory to understand how (and when) they work.

3/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

What RSHs Do We Consider?

Theoretically considered RSHs

- **(1+1) EA**
- $(1+\lambda)$ EA (offspring population)
- $(\mu+1)$ EA (parent population)
- $(\mu+1)$ GA (parent population and crossover)
- SEMO, DEMO, FEMO, ... (multi-objective)
- **Randomized Local Search (RLS)**
- Metropolis Algorithm/Simulated Annealing (MA/SA)
- Ant Colony Optimization (ACO)
- Particle Swarm Optimization (PSO)
- ...

First of all: define the simple ones

4/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

The Most Basic RSHs

(1+1) EA and RLS for maximization problems

(1+1) EA

- 1 Choose $x_0 \in \{0, 1\}^n$ uniformly at random.
- 2 For $t := 0, \dots, \infty$
 - 1 Create y by flipping each bit of x_t indep. with probab. $1/n$.
 - 2 If $f(y) \geq f(x_t)$ set $x_{t+1} := y$ else $x_{t+1} := x_t$.

RLS

- 1 Choose $x_0 \in \{0, 1\}^n$ uniformly at random.
- 2 For $t := 0, \dots, \infty$
 - 1 Create y by flipping one bit of x_t uniformly.
 - 2 If $f(y) \geq f(x_t)$ set $x_{t+1} := y$ else $x_{t+1} := x_t$.

5/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

What Kind of Theory Are We Interested in?

- **Not studied here:** convergence, local progress, models of EAs (e. g., infinite populations), ...
- Treat RSHs as randomized algorithm!
- Analyze their “runtime” (computational complexity) on selected problems

Definition

Let RSH A optimize f . Each f -evaluation is counted as a time step. The *runtime* $T_{A,f}$ of A is the random first point of time such that A has sampled an optimal search point.

- Often considered: expected runtime, distribution of $T_{A,f}$
- Asymptotical results w. r. t. n

6/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

How Do We Obtain Results?

We use (rarely in their pure form):

- Coupon Collector’s Theorem
- Concentration inequalities: Markov, Chebyshev, Chernoff, Hoeffding, ... bounds
- Markov chain theory: waiting times, first hitting times
- Rapidly Mixing Markov Chains
- Random Walks: Gambler’s Ruin, drift analysis, martingale theory, electrical networks
- Random graphs (esp. random trees)
- Identifying typical events and failure events
- Potential functions and amortized analysis
- ...

Adapt tools from the analysis of randomized algorithms; understanding the stochastic process is often the hardest task.

7/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Early Results

Analysis of RSHs already in the 1980s:

- Sasaki/Hajek (1988): SA and Maximum Matchings
- Sorkin (1991): SA vs. MA
- Jerrum (1992): SA and Cliques
- Jerrum/Sorkin (1993, 1998): SA/MA for Graph Bisection
- ...

High-quality results, but limited to SA/MA (nothing about EAs) and hard to generalize.

Since the early 1990s

Systematic approach for the analysis of RSHs, building up a completely new research area

8/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

This Tutorial

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for $(1+1)$ EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

9/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

How the Systematic Research Began — Toy Problems

Simple example functions (test functions)

- $\text{OneMax}(x_1, \dots, x_n) = x_1 + \dots + x_n$
- $\text{LeadingOnes}(x_1, \dots, x_n) = \sum_{i=1}^n \prod_{j=1}^i x_j$
- $\text{BinVal}(x_1, \dots, x_n) = \sum_{i=1}^n 2^{n-i} x_i$
- polynomials of fixed degree

Goal: derive first runtime bounds and methods

Artificially designed functions

- with sometimes really horrible definitions
- but for the first time these allow rigorous statements

Goal: prove benefits and harm of RSH components,
e. g., crossover, mutation strength, population size ...

10/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Agenda

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for $(1+1)$ EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

11/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Example: OneMax

Theorem (e. g., Droste/Jansen/Wegener, 1998)

The expected runtime of the RLS, $(1+1)$ EA, $(\mu+1)$ EA, $(1+\lambda)$ EA on ONEMAX is $\Omega(n \log n)$.

Proof by modifications of Coupon Collector's Theorem.

Theorem (e. g., Mühlenbein, 1992)

The expected runtime of RLS and the $(1+1)$ EA on ONEMAX is $O(n \log n)$.

Holds also for population-based $(\mu+1)$ EA and
for $(1+\lambda)$ EA with small populations.

12/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Proof of the $O(n \log n)$ bound

- Fitness levels: $L_i := \{x \in \{0, 1\}^n \mid \text{ONEMAX}(x) = i\}$
- (1+1) EA never decreases its current fitness level.
- From i to some higher-level set with prob. at least

$$\underbrace{\binom{n-i}{1}}_{\text{choose a 0-bit}} \cdot \underbrace{\left(\frac{1}{n}\right)}_{\text{flip this bit}} \cdot \underbrace{\left(1 - \frac{1}{n}\right)^{n-1}}_{\text{keep the other bits}} \geq \frac{n-i}{en}$$

- Expected time to reach a higher-level set is at most $\frac{en}{n-i}$.
- Expected runtime is at most

$$\sum_{i=0}^{n-1} \frac{en}{n-i} = O(n \log n). \quad \square$$

13/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Later Results Using Toy Problems

- Find the theoretically optimal mutation strength ($1/n$ for OneMax!).
- Bound the optimization time for linear functions ($O(n \log n)$).
- optimal population size (often 1!)
- crossover vs. no crossover \rightarrow Real Royal Road Functions
- multistarts vs. populations
- frequent restarts vs. long runs
- dynamic schedules
- ...

14/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

RSHs for Combinatorial Optimization

- Analysis of runtime and approximation quality on well-known combinatorial optimization problems, e. g.,
 - sorting problems (is this an optimization problem?),
 - **covering problems**,
 - cutting problems,
 - subsequence problems,
 - **traveling salesman problem**,
 - Eulerian cycles,
 - **minimum spanning trees**,
 - **maximum matchings**,
 - **scheduling problems**,
 - **shortest paths**,
 - ...
- We do not hope: to be better than the best problem-specific algorithms
- Instead: maybe reasonable polynomial running times
- In the following no fine-tuning of the results

15/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Agenda

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for (1+1) EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

16/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Minimum Spanning Trees:

- **Given:** Undirected connected graph $G = (V, E)$ with n vertices and m edges with positive integer weights.
- **Find:** Edge set $E' \subseteq E$ with minimal weight connecting all vertices.
- Search space $\{0,1\}^m$
- Edge e_i is chosen iff $x_i=1$
- Consider (1+1) EA

17/88

Fitness function:

- Decrease number of connected components, find minimum spanning tree.
- $f(s) := (c(s), w(s))$.
Minimization of f with respect to the lexicographic order.

18/88

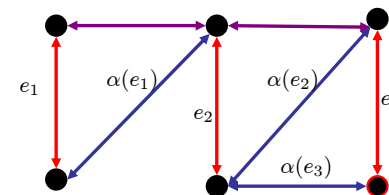
First goal: Obtain a connected subgraph of G .

How long does it take?

Connected graph in expected time $O(m \log n)$
(fitness-based partitions)

19/88

Bijection for minimum spanning trees:



$k := |E(T^*) \setminus E(T)|$
 Bijection $\alpha: E(T^*) \setminus E(T) \rightarrow E(T) \setminus E(T^*)$
 $\alpha(e_i)$ on the cycle of $E(T) \cup \{e_i\}$
 $w(e_i) \leq w(\alpha(e_i))$
 $\Rightarrow k$ accepted 2-bit flips that turn T into T^*

20/88

Upper Bound

Theorem:

The expected time until (1+1) EA constructs a minimum spanning tree is bounded by $O(m^2(\log n + \log w_{\max}))$.

Sketch of proof:

- $w(s)$ weight current solution s .
- w_{opt} weight minimum spanning tree T^*
- set of $m + 1$ operations to reach T^*
- $m' = m - (n - 1)$ 1-bit flips concerning non- T^* edges
 \Rightarrow spanning tree T
- k 2-bit flips defined by bijection
- $n - k$ non accepted 2-bit flips
- \Rightarrow average distance decrease $(w(s) - w_{\text{opt}})/(m + 1)$

21/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Proof

1-step (larger total weight decrease of 1-bit flips)

2-step (larger total weight decrease of 2-bit flips)

Consider 2-steps:

- Expected weight decrease by a factor $1 - (1/(2n))$
- Probability (n/m^2) for a good 2-bit flip
- Expected time until q 2-steps $O(qm^2/n)$

Consider 1-steps:

- Expected weight decrease by a factor $1 - (1/(2m'))$
- Probability (m'/m) for a good 1-bit flip
- Expected time until q 1-steps $O(qm/m')$

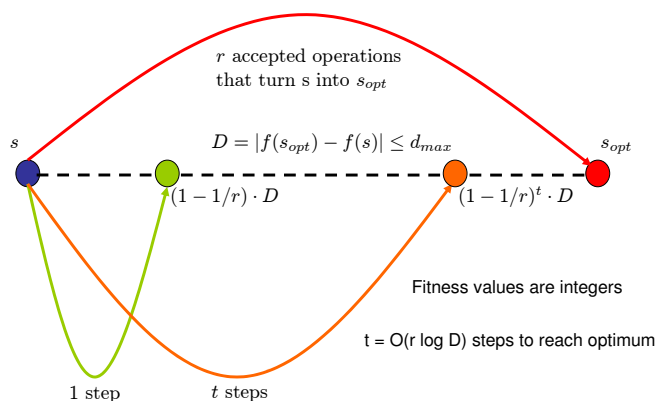
1-steps faster \Rightarrow show bound for 2-steps.

22/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Expected Multiplicative Distance Decrease (aka Drift Analysis)



23/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum distance: $w(s) - w_{\text{opt}} \leq D := m \cdot w_{\max}$

1 step: Expected distance at most $(1 - 1/(2n))(w(s) - w_{\text{opt}})$

t steps: Expected distance at most $(1 - 1/(2n))^t(w(s) - w_{\text{opt}})$

$t := \lceil 2 \cdot (\ln 2)n(\log D + 1) \rceil: (1 - 1/(2n))^t(w(s) - w_{\text{opt}}) \leq 1/2$

Expected number of 2-steps $2t = O(n(\log n + \log w_{\max}))$ (Markov)

Expected optimization time

$O(tm^2/n) = O(m^2(\log n + \log w_{\max}))$.

24/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Agenda

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for (1+1) EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

25/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum Matchings

A **matching** in an undirected graph is a subset of pairwise disjoint edges;
aim: find a maximum matching (solvable in poly-time)

Simple example: path of odd length



Maximum matching with more than half of edges

26/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum Matchings

A **matching** in an undirected graph is a subset of pairwise disjoint edges;
aim: find a maximum matching (solvable in poly-time)

Simple example: path of odd length



Suboptimal matching

Concept: augmenting path

- Alternating between edges being inside and outside the matching
- Starting and ending at “free” nodes not incident on matching
- Flipping all choices along the path improves matching

Example: whole graph is augmenting path

Interesting: how simple EAs find augmenting paths

26/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum Matchings: Upper Bound

Fitness function $f: \{0, 1\}^{\# \text{ edges}} \rightarrow \mathbb{R}$:

- one bit for each edge, value 1 iff edge chosen
- value for legal matchings: size of matching
- otherwise penalty leading to empty matching

Example: path with $n + 1$ nodes, n edges: bit string selects edges



Theorem

The expected time until (1+1) EA finds a maximum matching on a path of n edges is $O(n^4)$.

27/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum Matchings: Upper Bound (Ctnd.)

Proof idea for $O(n^4)$ bound

- Consider the level of second-best matchings.
- Fitness value does not change (walk on *plateau*).
- If “free” edge: chance to flip one bit! \rightarrow probability $\Theta(1/n)$.
- Else steps flipping two bits \rightarrow probability $\Theta(1/n^2)$.
- Shorten or lengthen augmenting path
- At length 1, chance to flip the free edge!



- Length changes according to a **fair random walk**
 \rightarrow equal probability for lengthenings and shortenings

28/88

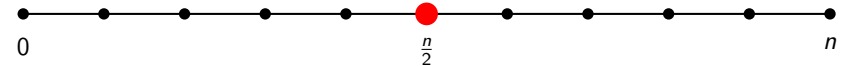
Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Fair Random Walk

Scenario: fair random walk

- Initially, player *A* and *B* both have $\frac{n}{2}$ USD
- Repeat: flip a coin
- If heads: *A* pays 1 USD to *B*, tails: other way round
- Until one of the players is ruined.



How long does the game take in expectation?

Theorem:

Fair random walk on $\{0, \dots, n\}$ takes in expectation $O(n^2)$ steps.

29/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum Matchings: Upper Bound (Ctnd.)

Proof idea for $O(n^4)$ bound

- Consider the level of second-best matchings.
- Fitness value does not change (walk on *plateau*).
- If “free” edge: chance to flip one bit! \rightarrow probability $\Theta(1/n)$.
- Else steps flipping two bits \rightarrow probability $\Theta(1/n^2)$.
- Shorten or lengthen augmenting path
- At length 1, chance to flip the free edge!



Length changes according to a **fair random walk**, expected $O(n^2)$ two-bit flips suffice, expected optimization time $O(n^2) \cdot O(n^2) = O(n^4)$.

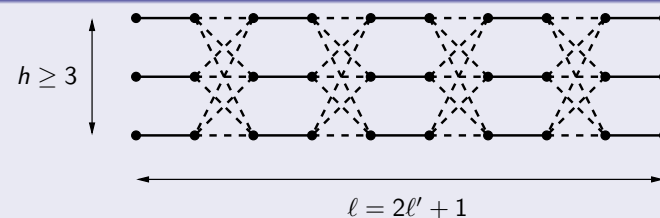
30/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Maximum Matchings: Lower Bound

Worst-case graph $G_{h,\ell}$



Augmenting path can get shorter **but is more likely to get longer.**
 (unfair random walk)

Theorem

For $h \geq 3$, $(1+1)$ EA has exponential expected optimization time $2^{\Omega(\ell)}$ on $G_{h,\ell}$.

Proof requires analysis of negative drift (simplified drift theorem).

31/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Insight: do not hope for exact solutions but for approximations

For maximization problems: solution with value a is called $(1 + \varepsilon)$ -approximation if $\frac{\text{OPT}}{a} \leq 1 + \varepsilon$, where OPT optimal value.

Theorem

For $\varepsilon > 0$, $(1+1)$ EA finds a $(1 + \varepsilon)$ -approximation of a maximum matching in expected time $O(m^{2/\varepsilon+2})$ (m number of edges).

Proof idea: If current solution worse than $(1 + \varepsilon)$ -approximate, there is a "short" augmenting path (length $\leq 2/\varepsilon + 1$); flip it in one go.

32/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for $(1+1)$ EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

33/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

All-pairs-shortest-path (APSP) problem

Given: Connected directed graph $G = (V, E)$, $|V| = n$ and $|E| = m$, and a function $w: E \rightarrow \mathbb{N}$ which assigns positive integer weights to the edges.

Compute from each vertex $v_i \in V$ a shortest path (path of minimal weight) to every other vertex $v_j \in V \setminus \{v_i\}$

Representation:

Individuals are paths between two particular vertices v_i and v_j

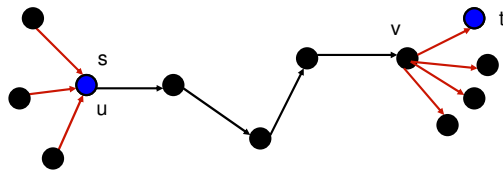
Initial Population: $P := \{I_{u,v} = (u, v) \mid (u, v) \in E\}$

Mutation-based EA

Mutation:

Pick individual $I_{u,v}$ uniformly at random

$E^-(u)$: incoming edges of u $E^+(v)$: outgoing edges of v



Pick uniformly at random an edge $e = (x, y) \in E^-(u) \cup E^+(v)$

Add e

New individual $I'_{s,t}$

Steady State EA

1. Set $P = \{I_{u,v} = (u, v) \mid (u, v) \in E\}$.
2. Choose an individual $I_{x,y} \in P$ uniformly at random.
3. Mutate $I_{x,y}$ to obtain an individual $I'_{s,t}$.
4. If there is no individual $I_{s,t} \in P$, $P = P \cup \{I'_{s,t}\}$,
else if $f(I'_{s,t}) \leq f(I_{s,t})$, $P = (P \cup \{I'_{s,t}\}) \setminus \{I_{s,t}\}$
5. Repeat Steps 2–4 forever.

Lemma:

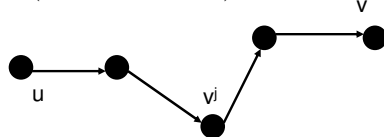
Let $\ell \geq \log n$. The expected time until has found all shortest paths with at most ℓ edges is $O(n^3 \ell)$.

Proof idea:

Consider two vertices u and v , $u \neq v$.

Let $\gamma := (v^1 = u, v^2, \dots, v^{\ell'+1} = v)$ be a shortest path from u to v consisting of ℓ' , $\ell' \leq \ell$, edges in G

the sub-path $\gamma' = (v^1 = u, v^2, \dots, v^j)$ is a shortest path from u to v^j .



$(E \setminus \{e\})$

Population size is upper bounded n^2
(for each pair of vertices at most one path)

- Pick shortest path from u to v_j and append edge (v_j, v_{j+1})
- Shortest path from u to v_{j+1}
- Probability to pick I_{u,v_j} is at least $1/n^2$
- Probability to append right edge is at least $1/(2n)$
- **Success** with probability at least $p = 1/(2n^3)$
- **At most 1 successes needed** to obtain shortest path from u to v

Analysis

Consider typical run consisting of $T = cn^3l$ steps.

What is the probability that the shortest path from u to v has been obtained?

We need at most l successes, where a success happens in each step with probability at least $p = 1/(2n^3)$

Define for each step i a random variable X_i .

$X_i = 1$ if step i is a success

$X_i = 0$ if step i is not a success

$$\text{Prob}(X_i = 1) \geq p = 1/(2n^3) \quad X = \sum_{i=1}^T X_i \quad X \geq \ell ???$$

$$\text{Expected number of successes } E(X) \geq T/(2n^3) = \frac{cn^3\ell}{2n^3} = \frac{c\ell}{2}$$

$$\text{Chernoff: } \text{Prob}(X < (1 - \delta)E(x)) \leq e^{-E(X)\delta^2/2}$$

$$\delta = \frac{1}{2}$$

$$\text{Prob}(X < (1 - \frac{1}{2})E(x)) \leq e^{-E(X)/8} \leq e^{-T/(16n^3)} = e^{-cn^3\ell/(16n^3)} = e^{-c\ell/(16)}$$

$$\text{Probability for failure of at least one pair of vertices at most: } n^2 \cdot e^{-c\ell/16}$$

c large enough and $\ell \geq \log n$:

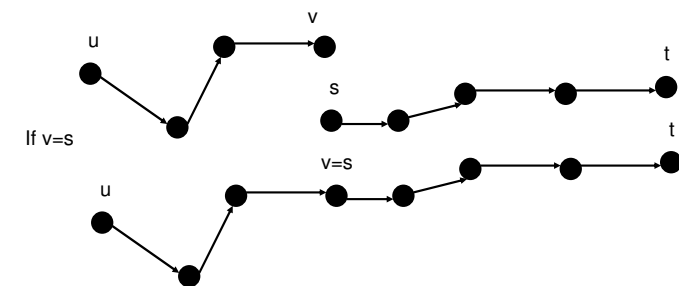
$$\text{No failure in any path with probability at least } \alpha = 1 - n^2 \cdot e^{-c\ell/16} = 1 - o(1)$$

Holds for any phase of T steps

$$\text{Expected time upper bound by } T/\alpha = O(n^3\ell)$$

Crossover

Pick two individuals $I_{u,v}$ and $I_{s,t}$ from population uniformly at random.



Theorem

The expected optimization time of Steady State EA for the APSP problem is $O(n^4)$.

Remark:

There are instances where the expected optimization of $(\mu + 1)$ -EA is $\Omega(n^4)$

Question:

Can crossover help to achieve a better expected optimization time?

Steady State GA

1. Set $P = \{I_{u,v} = (u,v) \mid (u,v) \in E\}$.
2. Choose $r \in [0, 1]$ uniformly at random.
3. If $r \leq p_c$, choose two individuals $I_{x,y} \in P$ and $I_{x',y'} \in P$ uniformly at random and perform crossover to obtain an individual $I'_{s,t}$, else choose an individual $I_{x,y} \in P$ uniformly at random and mutate $I_{x,y}$ to obtain an individual $I'_{s,t}$.
4. If $I'_{s,t}$ is a path from s to t then
 - ★ If there is no individual $I_{s,t} \in P$, $P = P \cup \{I'_{s,t}\}$,
 - ★ else if $f(I'_{s,t}) \leq f(I_{s,t})$, $P = (P \cup \{I'_{s,t}\}) \setminus \{I_{s,t}\}$.
5. Repeat Steps 2–4 forever.

p_c is a constant

Theorem:

The expected optimization time of Steady State GA is $O(n^{3.5} \sqrt{\log n})$.

Mutation and $\ell^* := \sqrt{n \log n}$

All shortest path of length at most ℓ^* edges are obtained

Show: Longer paths are obtained by crossover within the stated time bound.

Analysis Crossover

Long paths by crossover:

Assumption: All shortest paths with at most ℓ^* edges have already been obtained.

Assume that all shortest paths of length $k \leq \ell^*$ have been obtained.

What is the expected time to obtain all shortest paths of length at most $3k/2$?

Analysis Crossover

Consider pair of vertices x and y for which a shortest path of r , $k < r \leq 3k/2$, edges exists.

There are $2k-r$ pairs of shortest paths of length at most k that can be joined to obtain shortest path from x to y .

Probability for one specific pair: at least $1/n^4$

At least $2k+1-r$ possible pairs: probability at least $(2k+1-r)/n^4 \geq k/(2n^4)$

At most n^2 shortest paths of length r , $k < r \leq 3k/2$

Time to collect all paths $O(n^4 \log n / k)$
(similar to Coupon Collectors Theorem)

Analysis Crossover

Sum up over the different values of k , namely

$$\sqrt{n \log n}, c \cdot \sqrt{n \log n}, c^2 \cdot \sqrt{n \log n}, \dots, c^{\log_c(n/\sqrt{n \log n})} \cdot \sqrt{n \log n},$$

where $c = 3/2$.

Expected Optimization

$$\sum_{s=0}^{\log_c(n/\sqrt{n \log n})} \left(O\left(\frac{n^4 \log n}{\sqrt{n \log n}}\right) c^{-s} \right) = O(n^{3.5} \sqrt{\log n}) \sum_{s=0}^{\infty} c^{-s} = O(n^{3.5} \sqrt{\log n})$$

Agenda

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for (1+1) EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

Makespan Scheduling

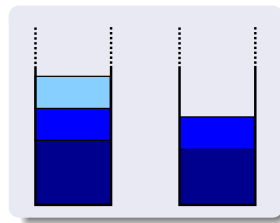
What about NP-hard problems? → Study approximation quality

Makespan scheduling on 2 machines:

- n objects with weights/processing times w_1, \dots, w_n
- 2 machines (bins)
- Minimize the total weight of fuller bin = makespan.

Formally, find $I \subseteq \{1, \dots, n\}$ minimizing

$$\max \left\{ \sum_{i \in I} w_i, \sum_{i \notin I} w_i \right\}.$$



Sometimes also called the **Partition** problem.

This is an “easy” NP-hard problem, good approximations possible

Fitness Function

- Problem encoding: bit string x_1, \dots, x_n reserves a bit for each object, put object i in bin $x_i + 1$.

- Fitness function

$$f(x_1, \dots, x_n) := \max \left\{ \sum_{i=1}^n w_i x_i, \sum_{i=1}^n w_i (1 - x_i) \right\}$$

to be minimized.

- Consider (1+1) EA and RLS.

Types of Results

- Worst-case results
- Success probabilities and approximations
- An average-case analysis
- A parameterized analysis

52/88

Frank Neumann, Carsten Witt

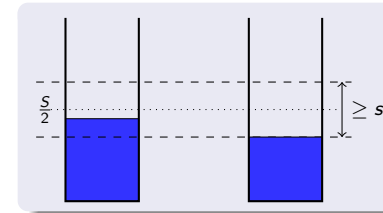
Bioinspired Computation in Combinatorial Optimization

Sufficient Conditions for Progress

Abbreviate $S := w_1 + \dots + w_n \Rightarrow$ perfect partition has cost $\frac{S}{2}$.

Suppose we know

- s^* = size of smallest object in the fuller bin,
 - $f(x) > \frac{S}{2} + \frac{s^*}{2}$ for the current search point x
- then the solution is improvable by a single-bit flip.



If $f(x) < \frac{S}{2} + \frac{s^*}{2}$, no improvements can be guaranteed.

Lemma

If smallest object in fuller bin is always bounded by s^ then $(1+1)$ EA and RLS reach f -value $\leq \frac{S}{2} + \frac{s^*}{2}$ in expected $O(n^2)$ steps.*

53/88

Frank Neumann, Carsten Witt

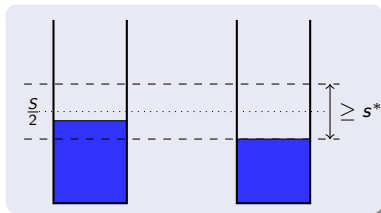
Bioinspired Computation in Combinatorial Optimization

Sufficient Conditions for Progress

Abbreviate $S := w_1 + \dots + w_n \Rightarrow$ perfect partition has cost $\frac{S}{2}$.

Suppose we know

- s^* = size of smallest object in the fuller bin,
 - $f(x) > \frac{S}{2} + \frac{s^*}{2}$ for the current search point x
- then the solution is improvable by a single-bit flip.



If $f(x) < \frac{S}{2} + \frac{s^*}{2}$, no improvements can be guaranteed.

Lemma

If smallest object in fuller bin is always bounded by s^ then $(1+1)$ EA and RLS reach f -value $\leq \frac{S}{2} + \frac{s^*}{2}$ in expected $O(n^2)$ steps.*

53/88

Frank Neumann, Carsten Witt

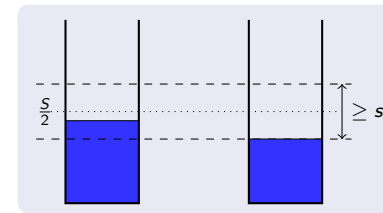
Bioinspired Computation in Combinatorial Optimization

Sufficient Conditions for Progress

Abbreviate $S := w_1 + \dots + w_n \Rightarrow$ perfect partition has cost $\frac{S}{2}$.

Suppose we know

- s^* = size of smallest object in the fuller bin,
 - $f(x) > \frac{S}{2} + \frac{s^*}{2}$ for the current search point x
- then the solution is improvable by a single-bit flip.



If $f(x) < \frac{S}{2} + \frac{s^*}{2}$, no improvements can be guaranteed.

Lemma

If smallest object in fuller bin is always bounded by s^ then $(1+1)$ EA and RLS reach f -value $\leq \frac{S}{2} + \frac{s^*}{2}$ in expected $O(n^2)$ steps.*

53/88

Frank Neumann, Carsten Witt

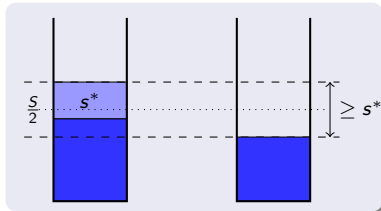
Bioinspired Computation in Combinatorial Optimization

Sufficient Conditions for Progress

Abbreviate $S := w_1 + \dots + w_n \Rightarrow$ perfect partition has cost $\frac{S}{2}$.

Suppose we know

- s^* = size of smallest object in the fuller bin,
 - $f(x) > \frac{S}{2} + \frac{s^*}{2}$ for the current search point x
- then the solution is improvable by a single-bit flip.



If $f(x) < \frac{S}{2} + \frac{s^*}{2}$, no improvements can be guaranteed.

Lemma

If smallest object in fuller bin is always bounded by s^* then $(1+1)$ EA and RLS reach f -value $\leq \frac{S}{2} + \frac{s^*}{2}$ in expected $O(n^2)$ steps.

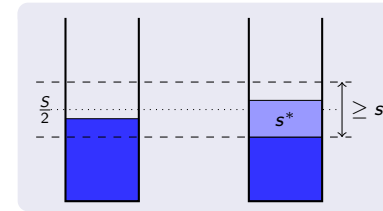
53/88

Sufficient Conditions for Progress

Abbreviate $S := w_1 + \dots + w_n \Rightarrow$ perfect partition has cost $\frac{S}{2}$.

Suppose we know

- s^* = size of smallest object in the fuller bin,
 - $f(x) > \frac{S}{2} + \frac{s^*}{2}$ for the current search point x
- then the solution is improvable by a single-bit flip.



If $f(x) < \frac{S}{2} + \frac{s^*}{2}$, no improvements can be guaranteed.

Lemma

If smallest object in fuller bin is always bounded by s^* then $(1+1)$ EA and RLS reach f -value $\leq \frac{S}{2} + \frac{s^*}{2}$ in expected $O(n^2)$ steps.

53/88

Worst-Case Results

Theorem

On any instance to the makespan scheduling problem, the $(1+1)$ EA and RLS reach a solution with approximation ratio $\frac{4}{3}$ in expected time $O(n^2)$.

Use study of object sizes and previous lemma.

Theorem

There is an instance W_ε^* such that the $(1+1)$ EA and RLS need with prob. $\Omega(1)$ at least $n^{\Omega(n)}$ steps to find a solution with a better ratio than $4/3 - \varepsilon$.

54/88

Worst-Case Instance

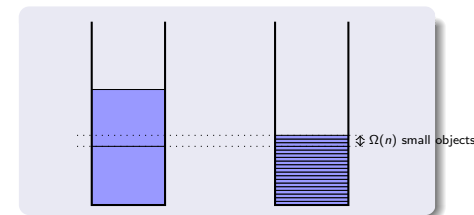
Instance $W_\varepsilon^* = \{w_1, \dots, w_n\}$ is defined by $w_1 := w_2 := \frac{1}{3} - \frac{\varepsilon}{4}$ (big objects) and $w_i := \frac{1/3 + \varepsilon/2}{n-2}$ for $3 \leq i \leq n$, ε very small constant; n even

Sum is 1; there is a perfect partition.

But if one bin with big and one bin with small objects: value $\frac{2}{3} - \frac{\varepsilon}{2}$.

Move a big object in the emptier bin \Rightarrow value $(\frac{1}{3} + \frac{\varepsilon}{2}) + (\frac{1}{3} - \frac{\varepsilon}{4}) = \frac{2}{3} + \frac{\varepsilon}{4}$!

Need to move $\geq \varepsilon n$ small objects at once for improvement: very unlikely.



With constant probability in this situation, $n^{\Omega(n)}$ needed to escape.

55/88

Worst Case – PRAS by Parallelism

Previous result shows: success dependent on big objects

Theorem

On any instance, the (1+1) EA and RLS with prob. $\geq 2^{-c \lceil 1/\varepsilon \rceil \ln(1/\varepsilon)}$ find a $(1 + \varepsilon)$ -approximation within $O(n \ln(1/\varepsilon))$ steps.

- $2^{O(\lceil 1/\varepsilon \rceil \ln(1/\varepsilon))}$ parallel runs find a $(1 + \varepsilon)$ -approximation with prob. $\geq 3/4$ in $O(n \ln(1/\varepsilon))$ parallel steps.
- Parallel runs form a polynomial-time randomized approximation scheme (PRAS)!

56/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Average-Case Analyses

Models: each weight drawn independently at random, namely

- 1 uniformly from the interval $[0, 1]$,
- 2 exponentially distributed with parameter 1 (i.e., $\text{Prob}(X \geq t) = e^{-t}$ for $t \geq 0$).

Approximation ratio no longer meaningful, we investigate:

discrepancy = absolute difference between weights of bins.

How close to discrepancy 0 do we come?

58/88

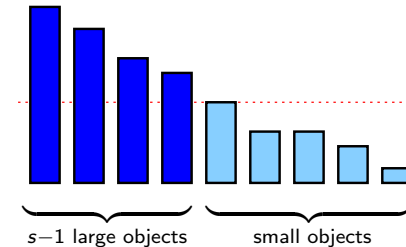
Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Worst Case – PRAS by Parallelism (Proof Idea)

Set $s := \lceil \frac{2}{\varepsilon} \rceil$

Assuming $w_1 \geq \dots \geq w_n$, we have $w_i \leq \varepsilon \frac{S}{2}$ for $i \geq s$.



analyze probability of distributing

- large objects in an optimal way,
- small objects greedily \Rightarrow error $\leq \varepsilon \frac{S}{2}$,

Random search rediscovers algorithmic idea of early algorithms.

57/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Makespan Scheduling – Known Average-Case Results

Deterministic, problem-specific heuristic LPT

Sort weights decreasingly,
put every object into currently emptier bin.

Known for both random models:

LPT creates a solution with discrepancy $O((\log n)/n)$.

What discrepancy do the (1+1) EA and RLS reach in poly-time?

59/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Average-Case Analysis of the (1+1) EA

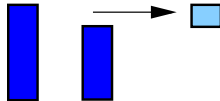
Theorem

In both models, the (1+1) EA reaches discrepancy $O((\log n)/n)$ after $O(n^{c+4} \log^2 n)$ steps with probability $1 - O(1/n^c)$.

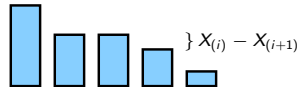
Almost the same result as for LPT!

Proof exploits order statistics:

If $X_{(i)}$ (i -th largest) in fuller bin, $X_{(i+1)}$ in emptier one, and discrepancy $> 2(X_{(i)} - X_{(i+1)}) > 0$, then objects can be swapped; discrepancy falls. Consider such “difference objects”.



W. h. p. $X_{(i)} - X_{(i+1)} = O((\log n)/n)$
(for $i = \Omega(n)$).



60/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

A Parameterized Analysis

Have seen: problem is hard for (1+1) EA/RLS in the worst case, but not so hard on average.

What **parameters** make the problem hard?

Definition

A problem is *fixed-parameter tractable (FPT)* if there is a problem parameter k such that it can be solved in time $f(k) \cdot \text{poly}(n)$, where $f(k)$ does not depend on n .

Intuition: for small k , we have an efficient algorithm.

Considered parameters (Sutton and Neumann, 2012):

- 1 Value of optimal solution
- 2 No. jobs on fuller machine in optimal solution
- 3 Unbalance of optimal solution

61/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Value of Optimal Solution

Recall **approximation** result: decent chance to distribute k big jobs optimally if k small.

Since $w_1 \geq \dots \geq w_n$, already $w_k \leq S/k$.

Consequence: optimal distribution of first k objects \rightarrow can reach makespan $S/2 + S/k$ by greedily treating the other objects.

Theorem

(1+1) EA and RLS find solution of makespan $\leq S/2 + S/k$ with probability $\Omega((2k)^{-ek})$ in time $O(n \log k)$. Multistarts have success probability $\geq 1/2$ after $O(2^{(e+1)k} k^{ek} n \log k)$ evaluations.

$2^{(e+1)k} k^{ek} \log k$ does not depend on $n \rightarrow$ a **randomized FPT-algorithm**.

62/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

No. Objects on Fuller Machine

Suppose: optimal solution puts only k objects on fuller machine.
Notion: k is called *critical path size*.

Intuition:

- Good chance of putting k objects on same machine if k small,
- other objects can be moved greedily.

Theorem

For critical path size k , multistart RLS finds optimum in $O(2^k (en)^{ck} n \log n)$ evaluations with probability $\geq 1/2$.

Due to term n^{ck} , result is somewhat weaker than FPT (a so-called XP-algorithm). Still, for constant k polynomial.

Remark: with (1+1)-EA, get an additional $\log w_1$ -term.

63/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Unbalance of Optimal Solution

Consider **discrepancy** of optimum $\Delta^* := 2(\text{OPT} - S/2)$.

Question/decision problem: Is $w_k \geq \Delta^* \geq w_{k+1}$?

Observation: If $\Delta^* \geq w_{k+1}$, optimal solution will put w_{k+1}, \dots, w_n on emptier machine. Crucial to distribute first k objects optimally.

Theorem

Multistart RLS with biased mutation (touches objects w_1, \dots, w_k with prob. $1/(kn)$ each) solves decision problem in $O(2^k n^3 \log n)$ evaluations with probability $\geq 1/2$.

Again, a randomized FPT-algorithm.

64/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Agenda

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for $(1+1)$ EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

65/88

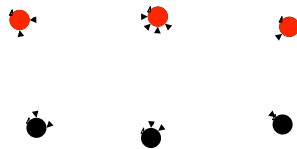
Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

The Problem

The Vertex Cover Problem:

Given an undirected graph $G=(V,E)$.



Find a minimum subset of vertices such that each edge is covered at least once.

NP-hard, several 2-approximation algorithms.

Simple single-objective evolutionary algorithms fail!!!

66/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

The Problem

Integer Linear Program (ILP)

$$\begin{aligned} \min & \sum_{i=1}^n x_i \\ \text{s.t. } & x_i + x_j \geq 1 \quad \forall \{i, j\} \in E \\ & x_i \in \{0, 1\} \end{aligned}$$

Linear Program (LP)

$$\begin{aligned} \min & \sum_{i=1}^n x_i \\ \text{s.t. } & x_i + x_j \geq 1 \quad \forall \{i, j\} \in E \\ & x_i \in [0, 1] \end{aligned}$$

Decision problem: Is there a set of vertices of size at most k covering all edges?

Our parameter: Value of an optimal solution (OPT)

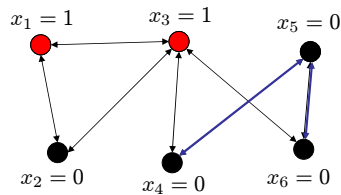
67/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Evolutionary Algorithm

Representation: Bitstrings of length n



Minimize fitness function:

$$f_1(x) = (|x|_1, |U(x)|)$$

$$f_1(x) = (2, 2)$$

$$f_2(x) = (|x|_1, LP(x))$$

$$f_2(x) = (2, 1)$$

$U(x)$: Edges not covered by x

$$G(x) = G(V, U(x))$$

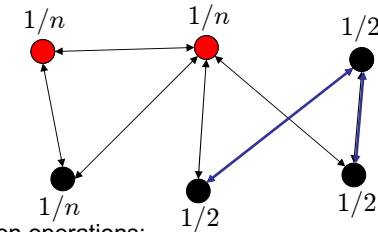
$LP(x)$: value of LP applied to $G(x)$

68/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Evolutionary Algorithm



Two mutation operations:

1. Standard bit mutation with probability $1/n$
2. Mutation probability $1/2$ for vertices adjacent to edges of $U(x)$.
Otherwise mutation probability $1/n$.

Decide uniformly at random which operator to use in next iteration

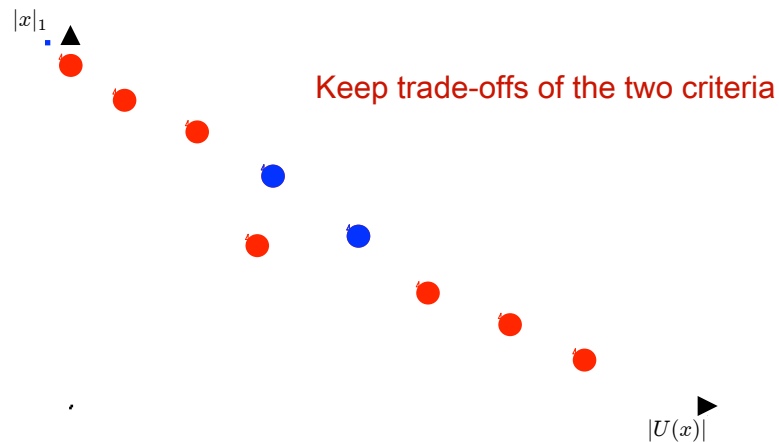
69/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Multi-Objective Approach:

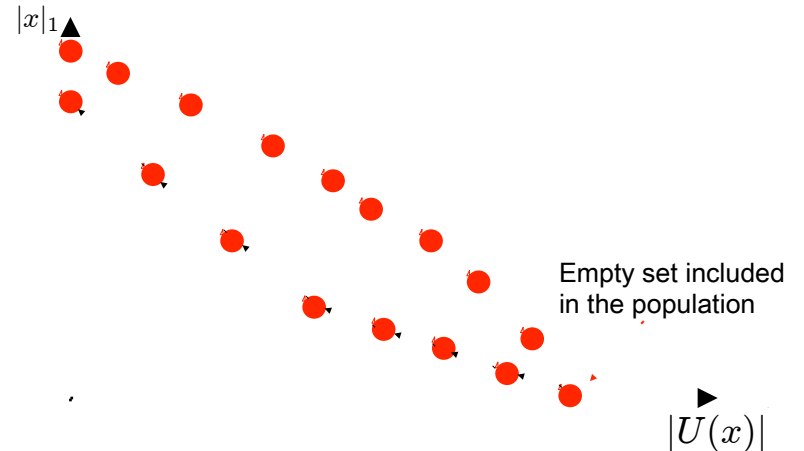
Treat the different objectives in the same way



70/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

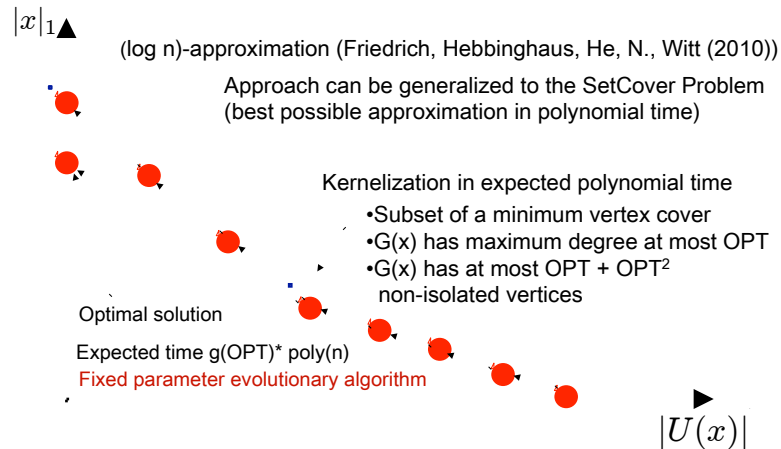


71/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

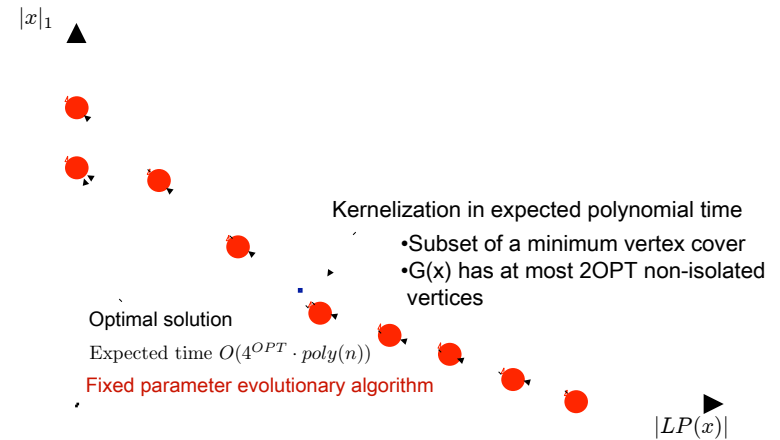
What can we say about these solutions?



72/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization



73/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Linear Programming

Combination with Linear Programming

- LP-relaxation is half integral, i.e.

$$x_i \in \{0, 1/2, 1\}, 1 \leq i \leq n$$

Theorem (Nemhauser, Trotter (1975)):

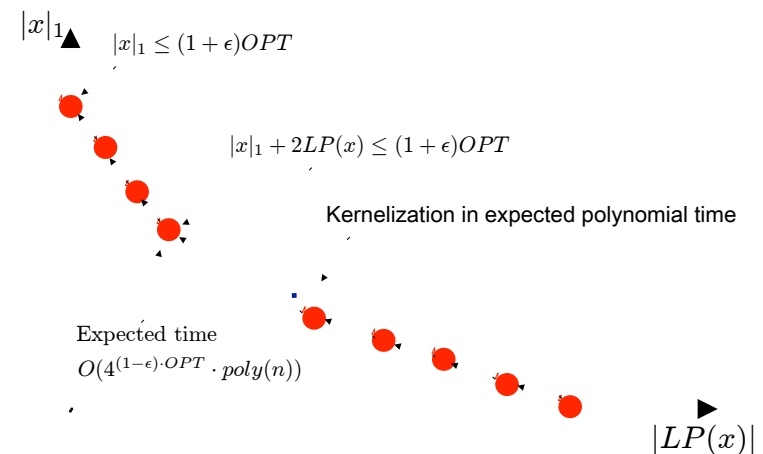
Let x^* be an optimal solution of the LP. Then there is a minimum vertex cover that contains all vertices v_i where $x_i^* = 1$.

Lemma:

All search points x with $LP(x) = LP(0^n) - |x|_1$ are Pareto optimal. They can be extended to minimum vertex cover by selecting additional vertices.

Can we also say something about approximations?

Approximations



75/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

- 1 The origins: example functions and toy problems
 - A simple toy problem: OneMax for (1+1) EA
- 2 Combinatorial optimization problems
 - Minimum spanning trees
 - Maximum matchings
 - Shortest paths
 - Makespan scheduling
 - Covering problems
 - Traveling salesman problem
- 3 End
- 4 References

76/88

Euclidean TSP

Given n points in the plane and Euclidean distances between the cities.

Find a shortest tour that visits each city exactly once and return to the origin.

NP-hard, PTAS, **FPT when number of inner points is the parameter.**

77/88

Representation and Mutation

Representation: Permutation of the n cities

For example: (3, 4, 1, 2, 5)

Inversion (inv) as mutation operator:

- Select i, j from $\{1, \dots, n\}$ uniformly at random and invert the part from position i to position j .
- Inv(2,5) applied to (3, 4, 1, 2, 5) yields (3, 5, 2, 1, 4)

78/88

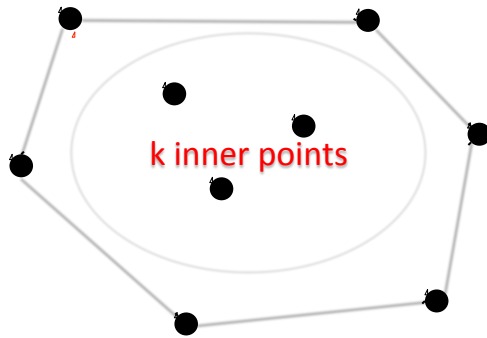
(1+1) EA

```
 $x \leftarrow$  a random permutation of  $[n]$ .  
repeat forever  
   $y \leftarrow \text{MUTATE}(x)$   
  if  $f(y) < f(x)$  then  $x \leftarrow y$ 
```

Mutation:

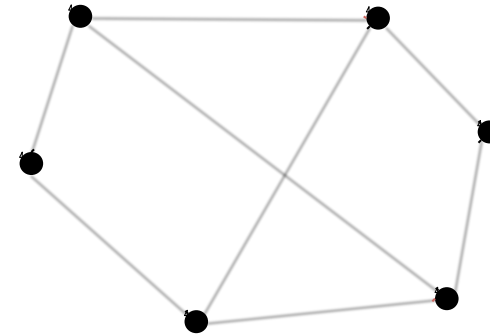
(1+1) EA: k random inversion,
 k chosen according to
1+Pois(1)

79/88



Convex hull containing n-k points

Intersection and Mutation

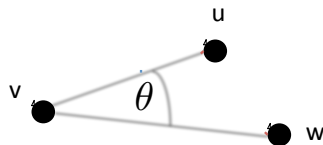


Angle bounded set of points

There may be an exponential number of inversion to end up in a local optimum if points are in arbitrary positions (Englert et al, 2007).

We assume that the set V is angle bounded

V is *angle-bounded* by $\epsilon > 0$ if for any three points $u, v, w \in V$, $0 < \epsilon < \theta < \pi - \epsilon$ where θ denotes the angle formed by the line from u to v and the line from v to w .



If V is angle-bounded then we get a lower bound on an improvement depending on ϵ

Progress

Assumptions:

d_{\max} : Maximum distance between any two points

d_{\min} : Minimum distance between any two points

V is angle-bounded by ϵ

Whenever the current tour is not intersection-free, we can guarantee a certain progress

Lemma:

Let x be a permutation such that is not intersection-free. Let y be the permutation constructed from an inversion on x that replaces two intersecting edges with two non-intersecting edges. Then, $f(x) - f(y) > 2d_{\min} \left(\frac{1 - \cos(\epsilon)}{\cos(\epsilon)} \right)$.

Tours

A tour x is either

- Intersection free
- Non intersection free

Intersection free tours are good. The points on the convex hull are already in the right order (Quintas and Supnick, 1965).

Claim: We do not spend too much time on non intersection free tours.

84/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Time spend on intersecting tours

Lemma:

Let $(x^{(1)}, x^{(2)}, \dots, x^{(t)}, \dots)$ denote the sequence of permutations generated by the $(1+1)$ -EA. Let α be an indicator variable defined on permutations of $[n]$ as

$$\alpha(x) = \begin{cases} 1 & x \text{ contains intersections;} \\ 0 & \text{otherwise.} \end{cases}$$

Then $E(\sum_{t=1}^{\infty} \alpha(x^{(t)})) = O\left(n^3 \left(\frac{d_{max}}{d_{min}} - 1\right) \left(\frac{\cos(\epsilon)}{1 - \cos(\epsilon)}\right)\right)$.

For an $m \times m$ grid:

For points on an $m \times m$ grid this bound becomes $O(n^3 m^5)$.

85/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

Summary and Conclusions

Parameterized Result

Lemma:

Suppose V has k inner points and x is an intersection-free tour on V . Then there is a sequence of at most $2k$ inversions that transforms x into an optimal permutation.

Theorem:

Let V be a set of points quantized on an $m \times m$ and k be the number of inner points. Then the expected optimisation time of the $(1+1)$ -EA on V is $O(n^3 m^5) + O(n^{4k} (2k - 1)!)$.

86/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

- Runtime analysis of RSHs in combinatorial optimization
 - Starting from toy problems to real problems
 - Insight into working principles using runtime analysis
 - General-purpose algorithms successful for wide range of problems
 - Interesting, general techniques
 - Runtime analysis of new approaches possible
- An exciting research direction.










Thank you!

87/88

Frank Neumann, Carsten Witt

Bioinspired Computation in Combinatorial Optimization

References

-  F. Neumann and C. Witt (2010):
Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity.
Springer.
-  A. Auger and B. Doerr (2011):
Theory of Randomized Search Heuristics – Foundations and Recent Developments.
World Scientific Publishing
-  F. Neumann and I. Wegener (2007):
Randomized local search, evolutionary algorithms, and the minimum spanning tree problem.
Theoretical Computer Science 378(1):32–40.
-  O. Giel and I. Wegener (2003):
Evolutionary algorithms and the maximum matching problem.
Proc. of STACS '03, LNCS 2607, 415–426, Springer
-  B. Doerr, E. Happ and C. Klein (2012):
Crossover can provably be useful in evolutionary computation.
Theoretical Computer Science 425:17–33.
-  C. Witt (2005):
Worst-case and average-case approximations by simple randomized search heuristics.
Proc. of STACS 2005, LNCS 3404, 44–56, Springer.
-  T. Friedrich, J. He, N. Hebbinghaus, F. Neumann and C. Witt (2010):
Approximating covering problems by randomized search heuristics using multi-objective models.
Evolutionary Computation 18(4):617–633.
-  S. Kratsch and F. Neumann (2009):
Fixed-parameter evolutionary algorithms and the vertex cover problem.
In *Proc. of GECCO 2009*, 293–300. ACM.
-  A. M. Sutton and F. Neumann (2012):
A parameterized runtime analysis of evolutionary algorithms for the Euclidean traveling salesperson problem.
Proc. of AAAI 2012.

88/88